

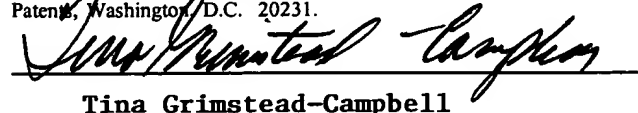
08957513-102497

APPENDIX G

"EXPRESS MAIL" Mailing Label Number EI267842785US

Date of Deposit October 24, 1997

I hereby certify under 37 CFR 1.10 that this correspondence is being deposited with the United States Postal Service as "Express Mail Post Office To Addressee" with sufficient postage on the date indicated above and is addressed to the Assistant Commissioner for Patents, Washington, D.C. 20231.


Tina Grimstead-Campbell

APPENDIX G

Byte Code Attributes Tables

Dividing Java byte codes into type groups

Each bytecode is assigned a 5 bit type associated with it. This is used to group the codes into similarly behaving sets. In general this behaviour reflects how the types of byte codes operate on the stack, but types 0, 13, 14, and 15 reflect specific kinds of instructions as denoted in the comments section.

The table below illustrates the state of the stack before and after each type of instruction is executed.

Type	Before execution	After exececutiion	Comment
0			Illegal instruction
1	stk0==int stk1==int	pop(1)	
2	stk0==int	pop(1)	
3	stk0==int stk1==int	pop(2)	
4			
5	push(1)		
6	stk0==int stk1==int	pop(3)	
7	stk0==int	pop(1)	
8	stk0==ref	pop(1)	
9	stk0==int	pop(1)	
10	push(1)	stk0<-int	
11	push(1)	stk0<-ref	
12	stk0==ref	stk0<-int	
13			DUPS, SWAP instructions
14			INVOKE instructions
15			FIELDS instructions
16		stk0<-ref	

Using Standard Java Byte Code (without reordering) - Attribute Lookup Table

```

/*
 * Table of bytecode decode information. This contains a bytecode type
 * and a bytecode length. We currently support all standard bytecodes
 * (ie. no quicks) which gives us codes 0 to 201 (202 codes in all).
 */

#define T_      0
#define T3      1
#define T6      2
#define T1      3
#define T2      4
#define T7      5
#define T9      6
#define T8      7
#define T12     8
#define T10     9
#define T5     10
#define T11     11
#define T16     12
#define T4      13
#define T13     14
#define T14     15
#define T15     16

#define D(T,L)                                     _BUILD_ITYPE_AND_ILENGTH(T, L)
#define _BUILD_ITYPE_AND_ILENGTH(T,L)             (_BUILD_ITYPE(T) | _BUILD_ILENGTH(L))
#define _BUILD_ITYPE(T)                           ((T) << 3)
#define _BUILD_ILENGTH(L)                         (L)
#define _GET_ITYPE(I)                             ((I) & 0xF8)
#define _GET_ILENGTH(I)                           ((I) & 0x07)

const uint8 _SCODE_decodeinfo[256] = {
    D( T4 , 1 ),      /* NOP */
    D( T11 , 1 ),     /* ACONST_NULL */
    D( T10 , 1 ),     /* ICONST_M1 */
    D( T10 , 1 ),     /* ICONST_0 */
    D( T10 , 1 ),     /* ICONST_1 */
    D( T10 , 1 ),     /* ICONST_2 */
    D( T10 , 1 ),     /* ICONST_3 */
    D( T10 , 1 ),     /* ICONST_4 */
    D( T10 , 1 ),     /* ICONST_5 */
    D( T_ , 1 ),
    D( T_ , 1 ),
    D( T_ , 1 ),
    D( T_ , 1 ),
    D( T_ , 1 ),
    D( T_ , 1 ),
    D( T10 , 2 ),     /* BIPUSH */
    D( T10 , 3 ),     /* SIPUSH */
    D( T_ , 2 ),     /* LDC1 */
    D( T11 , 3 ),     /* LDC2 */
    D( T_ , 3 ),
    D( T5 , 2 ),     /* ILOAD */
    D( T_ , 2 ),
    D( T_ , 2 ),
    D( T_ , 2 ),
    D( T5 , 2 ),     /* ALOAD */
    D( T5 , 1 ),     /* ILOAD_0 */
    D( T5 , 1 ),     /* ILOAD_1 */
    D( T5 , 1 ),     /* ILOAD_2 */
    D( T5 , 1 ),     /* ILOAD_3 */
    D( T_ , 1 ),
    D( T_ , 1 ),
    D( T_ , 1 ),
    D( T_ , 1 ),
    D( T_ , 1 )
}

```

G-2

08097512 100449

```
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T5 , 1 ),      /* ALOAD_0 */
D( T5 , 1 ),      /* ALOAD_1 */
D( T5 , 1 ),      /* ALOAD_2 */
D( T5 , 1 ),      /* ALOAD_3 */
D( T_ , 1 ),      /* IALOAD */
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),      /* AALOAD */
D( T7 , 1 ),      /* BALOAD */
D( T_ , 1 ),      /* CALOAD */
D( T7 , 1 ),      /* SALOAD */
D( T2 , 2 ),      /* ISTORE */
D( T_ , 2 ),
D( T_ , 2 ),
D( T8 , 2 ),      /* ASTORE */
D( T2 , 1 ),      /* ISTORE_0 */
D( T2 , 1 ),      /* ISTORE_1 */
D( T2 , 1 ),      /* ISTORE_2 */
D( T2 , 1 ),      /* ISTORE_3 */
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T8 , 1 ),      /* ASTORE_0 */
D( T8 , 1 ),      /* ASTORE_1 */
D( T8 , 1 ),      /* ASTORE_2 */
D( T8 , 1 ),      /* ASTORE_3 */
D( T_ , 1 ),      /* IASTORE */
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),      /* AASTORE */
D( T6 , 1 ),      /* BASTORE */
D( T_ , 1 ),      /* CASTORE */
D( T6 , 1 ),      /* SASTORE */
D( T2 , 1 ),      /* POP */
D( T3 , 1 ),      /* POP2 */
D( T13 , 1 ),      /* DUP */
D( T13 , 1 ),      /* DUP_X1 */
D( T13 , 1 ),      /* DUP_X2 */
D( T13 , 1 ),      /* DUP2 */
D( T13 , 1 ),      /* DUP2_X1 */
D( T13 , 1 ),      /* DUP2_X2 */
D( T13 , 1 ),      /* SWAP */
D( T1 , 1 ),      /* IADD */
D( T_ , 1 ),
D( T_ , 1 ),
D( T1 , 1 ),
D( T_ , 1 ),      /* ISUB */
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T1 , 1 ),      /* IMUL */
D( T_ , 1 ),
D( T_ , 1 ),
```

9-3

```

D( T_ , 1 ),
D( T1 , 1 ),          /* IDIV          */
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T1 , 1 ),          /* IREM          */
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T9 , 1 ),          /* INEG          */
D( T_ , 1 ),
D( T_ , 1 ),
D( T1 , 1 ),          /* ISHL          */
D( T_ , 1 ),
D( T1 , 1 ),          /* ISHR          */
D( T_ , 1 ),
D( T1 , 1 ),          /* IUSHR         */
D( T_ , 1 ),
D( T1 , 1 ),          /* IAND          */
D( T_ , 1 ),
D( T1 , 1 ),          /* IOR           */
D( T_ , 1 ),
D( T1 , 1 ),          /* IXOR          */
D( T_ , 1 ),
D( T4 , 3 ),          /* IINC          */
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T9 , 1 ),          /* INT2BYTE      */
D( T9 , 1 ),          /* INT2CHAR      */
D( T_ , 1 ),          /* INT2SHORT     */
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T2 , 3 ),          /* IFEQ          */
D( T2 , 3 ),          /* IFNE          */
D( T2 , 3 ),          /* IFLT          */
D( T2 , 3 ),          /* IFGE          */
D( T2 , 3 ),          /* IFGT          */
D( T2 , 3 ),          /* IFLT          */
D( T3 , 3 ),          /* IF_ICMPEQ     */
D( T3 , 3 ),          /* IF_ICMPNE     */
D( T3 , 3 ),          /* IF_ICMPLT     */
D( T3 , 3 ),          /* IF_ICMPGE     */
D( T3 , 3 ),          /* IF_ICMPGT     */
D( T3 , 3 ),          /* IF_ICMPLE     */
D( T3 , 3 ),          /* IF_ACMPEQ     */
D( T3 , 3 ),          /* IF_ACMPLT     */
D( T3 , 3 ),          /* IF_ACMPEQ     */
D( T3 , 3 ),          /* IF_ACMPLT     */
D( T4 , 3 ),          /* GOTO          */
D( T_ , 3 ),          /* JSR           */
D( T_ , 2 ),          /* RET           */
D( T2 , 0 ),          /* TABLESWITCH */
D( T2 , 0 ),          /* LOOKUPSWITCH*/
D( T2 , 1 ),          /* IRETURN       */
D( T_ , 1 ),
D( T_ , 1 ),
D( T_ , 1 ),
D( T8 , 1 ),          /* ARETURN       */
D( T4 , 1 ),          /* RETURN        */

```

G-4

D(T_ , 1),
D(T_ , 1),
D(T_ , 1),
D(T_ , 1),
D(T_ , 1),
D(T_ , 1),
D(T_ , 1),

};

11/11/11 11:11:11